1/3/1 (Item 1 from file: 351)
DIALOG(R)File 351:Derwent WPI

014956079 **Image available**
WPI Acc No: 2003-016593/200301
Related WPI Acc No: 2002-706384; 2004-355365
XRPX Acc No: N03-012512

Arithmetic coding apparatus selects current probability estimate or estimate after updating, based on state of arithmetic coding calculator whose output block is fixed to preset size

Patent Assignee: FUJITEC CO LTD (FUJI-N); MATSUSHITA GRAPHIC COMMUNICATION SYSTEMS (MATY); KYUSHU MATSUSHITA DENKI KK (MATU); PANASONIC COMMUNICATIONS CO LTD (MATU)
Inventor: HORIE H
Number of Countries: 028 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20020114529	A1	20020822	US 200246764	A	20020117	200301	B
EP 1235353	A2	20020828	EP 20021349	A	20020118	200301	
EP 1235436	A2	20020828	EP 20021523	A	20020122	200301	
JP 2003209699	A	20030725	JP 20022818	A	20020109	200351	
US 6677869	B2	20040113	US 200246764	A	20020117	200405	

Priority Applications (No Type Date): JP 20022818 A 20020109; JP 200147068 A 20010222
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020114529	A1		33	G06K-009/36	
EP 1235353	A2	E		H03M-007/40	
	Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR				
EP 1235436	A2	E		H04N-007/30	
	Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR				
JP 2003209699	A		20	H04N-001/413	
US 6677869	B2			G06K-009/36	

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-209699
(43)Date of publication of application : 25.07.2003

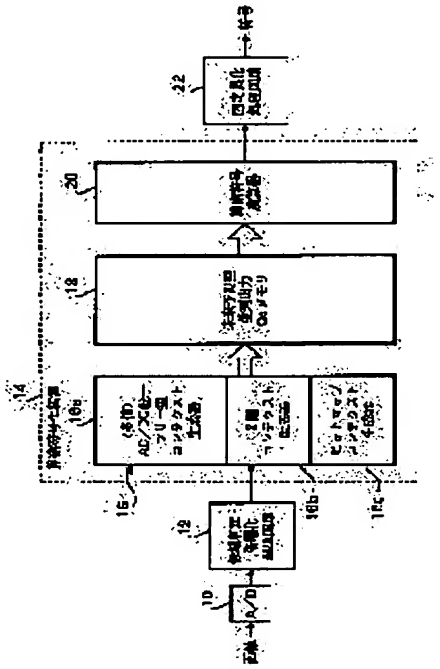
(51)Int.Cl. H04N 1/413
H03M 7/30
H03M 7/40
H04N 7/30

(21)Application number : 2002-002818
(22)Date of filing : 09.01.2002

(71)Applicant : PANASONIC COMMUNICATIONS CO LTD
(72)Inventor : HORIE HITOSHI

(54) ARITHMETIC CODER AND IMAGE PROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To realize arithmetic encoding and decoding of an ultra-high speed and a high compression ratio regardless of binary data or multi-value data, and also to simplify subsequent operation by making the quantities of arithmetic code uniform in each block.
SOLUTION: An image region judging/layering circuit 12 analyzes an input image, accurately reads its image information and arithmetically encodes it at an ultra-high speed. Disturbance in pipeline involved in generation of normalizing operation can be resolved by employing a future predicting parallel output Qe memory 18. At the time of encoding a multi-value image, a context generator 16 generates a common context without discrimination between AC and DC components in a DCT coefficient. Further, a fixed length source coding circuit 22 is provided to convert an arithmetic code having a variable length code to a code having a fixed length and to output it, thus facilitating editing of a reconstructed image.



LEGAL STATUS

[Date of request for examination] 24.09.2004

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2003－209699
（P2003－209699A）
(43)公開日　平成15年７月25日(2003.7.25)

(51)Int.Cl.⁷		識別記号	FI			テーマコード（参考）
H04N	1/413		H04N	1/413	Z	5C059
H03M	7/30		H03M	7/30	A	5C078
	7/40			7/40		5J064
H04N	7/30		H04N	7/133	Z	

審査請求　未請求　請求項の数９　ＯＬ　（全 20 頁）

(21)出願番号　特願2002－2818(P2002－2818)

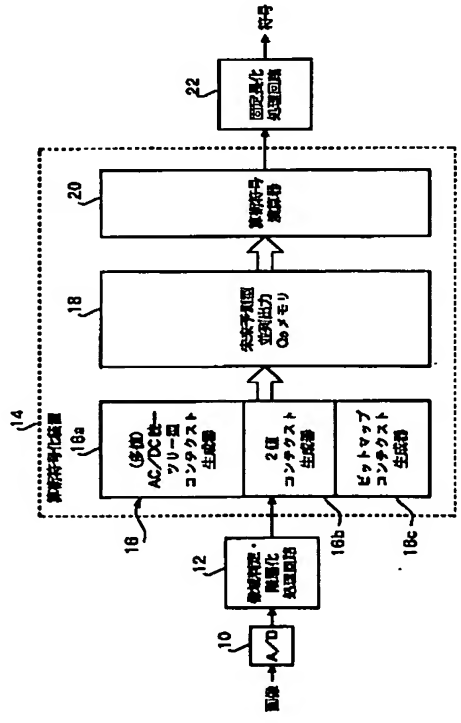
(22)出願日　平成14年１月９日(2002.1.9)

(71)出願人　597000489
パナソニック　コミュニケーションズ株式会社
福岡県福岡市博多区美野島四丁目１番62号
(72)発明者　堀江　等
東京都目黒区下目黒２丁目３番８号　松下電送システム株式会社内
(74)代理人　100105050
弁理士　鷲田　公一

最終頁に続く

(54)【発明の名称】　算術符号化装置および画像処理装置

(57)【要約】
【課題】　2値データ・多値データを問わず、超高速かつ高圧縮率の算術符号化・復号化を実現し、併せて、ブロック毎の算術符号量を一定値に揃えて、その後の処理を簡単化すること。
【解決手段】　像域判定・階層化処理回路12にて、入画画像を分析して正確に画像情報を読み取り、超高速に算術符号化する。正規化処理の発生に伴うパイプラインの乱れは、未来予測型並列出力Ｑｅメモリ18を採用することで解決する。また、コンテクスト生成器16において、多値画像符号化の際、DCT係数のAC成分とDC成分を区別することなく共通のコンテクストを発生させる。また、可変長符号である算術符号を、固定長化して出力するための固定長化処理回路22を設け、復元画像の編集を容易化する。



1

【特許請求の範囲】

【請求項１】　既に符号化済みのシンボル系列の状態（コンテクスト）から符号化シンボルの生成確率を推定し、その確率推定値および前記シンボルの予測値を符号器に供給して符号化を行う算術符号化装置であって、

中間調画像データに対して、所定サイズのブロックを単位として離散コサイン変換（ＤＣＴ）を行い、ＤＣＴ係数を得るＤＣＴ変換部と、

ＤＣ成分のＤＣＴ係数とＡＣ成分のＤＣＴ係数とを区別せずにコンテクストを生成するコンテクスト生成器と、

一つのコンテクスト毎に、前記シンボルの推定値およびその推定値に対応する状態番号を記憶しているコンテクストメモリと、

現在の確率推定値と、所定の状況が生じて更新がなされた後の確率推定値とを並列に出力する確率推定メモリと、

算術符号化演算を行う算術符号演算器と、

前記算術符号演算器の状態に応じて、前記現在の確率推定値、または、前記所定の状況が生じて更新がなされた後の確率推定値のいずれかを選択して前記算術符号演算器に供給するセレクタと、

前記算術符号演算器から出力される、前記ブロック毎の符号長を、強制的に所定のサイズに揃える固定長化部と、

を有することを特徴とする算術符号化装置。

【請求項２】　請求項１において、

前記コンテクスト生成器におけるコンテクストの生成処理、前記確率推定メモリの読み出し処理、前記コンテクストメモリの読み出し処理、および前記算術符号演算器における算術符号化演算の各々の処理を、同じサイクルで実行してパイプライン化し、１サイクルで１画素の符号を出力することを特徴とする算術符号化装置。

【請求項３】　請求項１において、

前記固定長化部は、

前記ブロックの各々の符号化に先立ってゼロクリアされる、前記算術符号演算器から出力される符号を一時的に蓄積する符号メモリと、

前記算術符号演算器から出力される符号の積算量が、所定のサイズと一致しているかを検出し、積算量が符号のサイズに達したときに、前記算術符号演算器に、一つのブロックについての算術符号化の終了を通知する符号長カウンタと、

前記符号メモリから、前記所定サイズ分の符号を読み出す符号読出し部と、

を有することを特徴とする算術符号化装置。

【請求項４】　請求項１において、

さらに、

符号化対象の画像データについて、タイルを単位として画像の種類を判定し、所定の種類のタイルに含まれる全画素を１画素毎に複数の階層に分類し、その階層の分類

2

に応じて、その画素のデータを前記ＤＣＴ変換部に供給するか、あるいは、その画素に対して所定の信号処理を施し、その処理後のデータを前記コンテクスト生成器に供給する画像階層化部、を有することを特徴とする算術符号化装置。

【請求項５】　請求項１記載の算術符号化装置を具備することを特徴とする画像処理装置。

【請求項６】　連続階調画像について、予め定めた大きさのブロック毎に画像情報を抽出する画像情報抽出部と、

抽出された画像情報を算術符号化する算術符号器と、

前記算術符号器から出力される符号を一時記憶するメモリと、

前記メモリから予め定めた量の符号を読み出すことで、前記ブロック毎の符号量を一定にする読出し部と、

を具備することを特徴とする算術符号化装置。

【請求項７】　請求項６において、

前記算術符号器は、一つのブロックの符号量が所定を超えたときに一つのブロックの終りを示すシンボルを符号化して、符号化処理を終了とすることを特徴とする算術符号化装置。

【請求項８】所定の数の符号を一時的に記憶するメモリと、

算術復号器と、

前記算術復号器へ入力される符号の数を計数する計数回路と、

前記算術復号器に、前記符号またはゼロ信号のいずれを与えるかを選択するためのセレクタと、を有し、

前記算術復号器への入力符号数が予め定めた所定の数を超えたならば、前記セレクタの出力を、符号からゼロ信号に切り替えることを特徴とする算術復号化装置。

【請求項９】　カラー画像を予め定めたブロックに分割する分割部と、

色空間の変換部と、

色成分ごとに、前記ブロックの符号バイト数を設定する、符号バイト数設定部と、

前記ブロックを、一定符号長で符号化する算術符号器と、

を有することを特徴とするカラー画像処理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、算術符号化装置および画像処理装置に関する。

【０００２】

【従来の技術】符号化シンボルを既に符号化済みの周辺画素の状態によって予測し、予測結果をその状態毎に定まる符号化シンボルの確率推定値に基づき算術符号化する方式は、圧縮率の点から最も優れた特性を示すことが知られている。

【０００３】JBIG(ITU勧告T. 82)に採用されている符号

器であるQM-coderは、算術符号化を行う装置の代表例である。

【０００４】以下、2値画像の算術符号器であるQM-coderの、一般的な構成と動作について説明する。

【０００５】QM-coderは、図18に示すように、コンテクスト生成部200と、コンテクストテーブル（コンテクストメモリ）210と、確率推定部220と、算術符号器230と、をもつ。

【０００６】コンテクスト生成部200は、符号化画素の周辺10画素によって作られる1024個の状態を検出する。図20にテンプレートの一例を示す。

【０００７】図中、“？”で示されるのが符号化対象の画素であり、また、“×”で示される10個の画素が参照画素である。一つの画素の符号化が終了すると、図20に点線で示されるように、テンプレートを右側に一つずらし、次の画素の符号化を行なう。

【０００８】10個の画素の値により決定される1024個の状態の各々は、コンテクスト(以下“Ｓ”と表す)と呼ばれる。コンテクスト毎に、優勢シンボルの予測値MPS(s)（すなわち、着目する符号化シンボルについて、MPSが“１”であると予測されれば、MPS（S）＝１である）と、確率推定器の状態番号とが、コンテクストメモリから読み出され、確率推定部220に出力される。

【０００９】確率推定器220は、これらの情報から劣勢シンボルの領域幅Qe(s)を算術符号器230に出力する。ここで“Qe”は、LPSが生起される確率であり、本明細書では、これを符号化シンボルの生起確率とか、単に、確率推定値という場合もある。

【００１０】また、劣勢シンボルの領域幅Qe（s）は、LPSの生起確率にオージェンドの幅をかけて算出される、LPSの生起確率に対応する幅を意味する。オージェンドとは、図19に示されるような、数直線（各選択区間）の全体の幅をいう。

【００１１】算術符号器230は、符号化シンボル、優勢シンボルの予測値MPS(s)および領域幅Qe(s)から算術符号化演算を実行し、符号を出力する。

【００１２】図19に示すように、算術符号化では、初期値0～1の数直線を優勢シンボル（MPS）の領域幅と劣勢シンボル（LPS）の領域幅に分ける。符号化対象のシンボル系列は、分割された領域内の代表点に対応させる。代表点は、部分区間内の一番下にとられる。

【００１３】符号化シンボルと予測値が同じときは、次のシンボルの符号化にはMPS幅が選ばれ、そうでなければLPS幅が選ばれる。

【００１４】上述のとおり、この領域幅の中に代表点を設けて、その代表点の2進小数点が符号を表わす。

【００１５】算術符号化演算では、領域幅が所定値未満になった時には、少数点の精度を防ぐために所定値（具体的には初期値の1/2）以上になるまで2倍処理を繰り返す。この処理を正規化処理という。

【００１６】また、正規化処理は、LPSを符号化したときも行われる。すなわち、推定がはずれてLPS幅が選択されると、そのLPS幅は、必ず、初期値の1/2より小さいため、毎回、正規化がなされることになる。

【００１７】正規化処理が行なわれる場合には、図18のコンテクストテーブル210における、MPS値や状態番号（ST）が更新される。状態番号の更新は、確率推定部220に書かれている“次の状態番号”が、コンテクストテーブル210にオーバーライトされることにより実現される。図18では、このオーバーライトを矢印ＲＸで示している。

【００１８】このコンテクストテーブル210の更新により、次もまた、前回と同じコンテクストであった（すなわち、図20のテンプレートを右に一つずらしても、参照画素の１と０の配置が前回と同じであった）としても、発生するQe（S）の値が異なることになる。

【００１９】これによって、より情報源の確率分布に適した値が選択されるようになる。つまり、符号化対象の画像への適応化がなされる。

【００２０】なお、符号化対象は、2値データに限られるものではない。中間調画像のデータのような多値データも、QM-coderで符号化することができる。但し、使用するコンテクストは、符号化対象に適合しているものを使わなければならない。

【００２１】例えば、離散コサイン変換係数（ＤＣＴ係数）には、DC成分とAC成分が含まれる。各成分は信号の性質が異なる。よって、それらに適したコンテクストモデルを構築することが、符号化効率を向上させる点では重要である。

【００２２】現状の技術では、DC成分とAC成分の各々に対して、別々にコンテクストが生成されている。

【００２３】

【発明が解決しようとする課題】従来の算術符号化処理では、以下のような課題がある。

【００２４】（課題１）本来、算術符号化は、圧縮率には優れているが、１シンボル毎にコンテクストの生成と、シンボルの生起確率情報の推定と、算術符号演算を繰り返す必要があることから、処理時間が長いという弱点がある。処理スピードの低下は、高画質かつ高速性が要求される、デジタルコピー機のような分野では、かなり大きな問題となる。

【００２５】また、算術符号処理をパイプライン化した場合、途中で正規化処理が発生すると、パイプラインに乱れが生じ、その結果として無駄な待ち時間が増える。よって処理効率が低下する場合がある。特に、正規化がかなりの頻度で発生し、かつ、コンテクストに連続性がある画像パターンでパイプラインの乱れが発生しやすいと考えられる。なお、復号化の場合にも同様の問題が生じる。

【００２６】よって、パイプライン化したとしても、そ

の精度は高いとはいえず、算術符号・復号処理の高速化は、あまり望めない。

【００２７】（課題２）2値データの符号化のみならず、多値データ（例えば、JPEG圧縮により得られる直交変換係数などの数値データ）を符号化、復号化する場合にも、高精度なパイプライン処理を行うようにするのが望ましい。

【００２８】しかし、直交変換係数などの数値を高速に算術符号化、復号化することに関して次の課題がある。

【００２９】つまり、パイプライン処理を有効に働かせるには、一定速度で次のパイプラインステージに必要なデータを連続的に供給しなければならない。

【００３０】例えば、復号化処理を高速に実行するためには、コンテクストインデックスは、復元される可能性のある複数シンボルに対して同時に供給する必要がある。このコンテクストインデックスのセットを、“コンテクストインデックスベクトル”と呼ぶことにする。

【００３１】このようなベクトル形式のコンテクストデータを一定速度で供給するのは、実際には困難である。上述のとおり、直交変換係数にはDC成分とAC成分とが含まれる。そして、DC成分とAC成分とは統計的性質が異なるので、現在の技術では、各成分毎に別々のコンテクストを設けている。

【００３２】異なるコンテクストを連続して供給するのは困難である。よって、符号化するべき複数のシンボルの中に、DC成分とAC成分が混在する場合、コンテクストインデックスベクトルを一定速度で供給することは困難である。

【００３３】したがって、多値データの高速な算術符号化は、2値データの場合以上に、困難である。

【００３４】（課題３）入力画像を所定サイズのブロック毎に分け、各ブロック単位で独立に算術符号化を行い、各ブロック単位で自由に復号することができると便利である。

【００３５】しかし、算術符号は可変長符号であるため、１ブロックに対応する符号の符号長は一定ではない。よって、算術符号化された画像データを復元する際に、一部のブロックのみについて画像の向きを変える処理を行う場合などには、対象となるブロックの先頭の画素のデータを探し出すのが困難である。

【００３６】本発明は、このような現状を考慮し、2値データ・多値データを問わず、超高速かつ高圧縮率の算術符号化・復号化を実現し、併せて、ブロック毎の算術符号量を一定値に揃えて、その後の処理を簡単化することを目的とする。

【００３７】

【課題を解決するための手段】（１）本発明の算術符号化装置では、符号化パラメータの更新が必要になった場合に、そのパラメータが更新された後に出力されるであろう確率推定値（未来の確率推定値）を、現在の推定値（通常の処理で出力される確率推定値）と共に、並列に出力する。そして、所定の状況が発生した場合（例えば、パラメータの更新の必要が生じ、かつ、コンテクストが連続するためにRAMの読み出しと書き込みが競合するような場合）には、現在の推定値ではなく、未来の推定値の方を符号器に供給する。これにより、ループを回してパラメータを更新している間、符号化を待つ必要がなくなり、パイプラインの乱れが防止される。

【００３８】（２）また、本発明では、圧縮率を絶対的に重要視する従来の固定的な考えを捨てて、処理スピードを最も重視する新規な考え方を導入する。この考え方に立脚して、DC成分用のコンテクストおよびAC成分用のコンテクストを積極的に共通にする。そして、コンテクストインデックスの生成を符号化も復号化も状態遷移テーブルで実現できるようにする。

【００３９】復号時のコンテクストインデックスベクトルの生成も容易となり、パイプライン処理に適した符号器や復号器が実現できる。符号化コンテクストを簡略化したぶん、圧縮性能が多少は低下する。しかし、本発明では、像域判定や階層化処理などにより、入力画像の情報を正確に分析し、高精度な圧縮を行うため、問題はまったく生じない。また、現実には、画質よりも処理速度が優先される場合も多く、実用上の問題は何もない。

【００４０】本発明によれば、2値画像および多値画像の双方について、柔軟に、しかも、ほとんど算術符号化アルゴリズムで決まる限界の速度でもって算術符号化・算術復号化することができる。

【００４１】しかも、2値画像と多値画像を区別することなく、共通のコンテクストを用いて符号・復号化ができるため、符号・復号器の構成もきわめて簡素化される。

【００４２】（３）また、算術符号を一旦、メモリにストアし、そのメモリから、常に所定のバイトのデータを読み出すことにより、１ブロックの符号を固定長化することができる。

【００４３】これにより、ブロックを単位として画像を部分的に復元する等の処理を、簡単に行うことができるようになる。

【００４４】

【発明の実施の形態】以下、本発明の実施の形態について、図面を参照して説明する。

【００４５】本発明の算術符号化装置の一例の全体構成の概要を図1に示す。

【００４６】図示されるように、本発明の算術符号化装置は、画像信号をA/D変換器10によりディジタルデータに変換した後、まず、像域判定・階層化処理回路12にて、像域判定ならびに階層化処理を行う。

【００４７】像域判定は、例えば、タイル（マクロブロック：例えば、32画素×32画素）を単位として行われる。

【００４８】階層化処理は、例えば、一つのタイルに含まれる全画素を、前景（FG）と背景（BG）に分類する処理である。このような処理により、入力された画像がもつ情報を正確に取得することができる。

【００４９】取得された画像情報は、完全にパイプライン化された算術符号化装置14にて、算術符号化される。

【００５０】算術符号装置では、予測値が実際の値と異なっていた場合に、そのペナルティーとして符号が出力され、コンテクストRAMの書き換えが行われる。したがって、そのコンテクストRAMの書き換えが行われるときには、必ず、パイプラインが乱れてしまう。

【００５１】本発明では、この問題点を、未来予測値の並列出力方式を採用すること、ならびに、DCT係数のAC成分/DC成分を区別せずに共通のコンテクストを使用することにより、克服する。

【００５２】これにより、ハザードレス・完全パイプライン処理が実現される。これにより、１クロックに同期して、１画素分の符号を連続的に生成することができる。

【００５３】算術符号化装置14は、コンテクスト生成器16（コンテクストRAMも含むものとする）と、未来予測型並列出力Ｑｅメモリ18と、算術符号演算器20とを含む。

【００５４】コンテクスト生成器16は、多値データ用コンテクスト生成において、AC成分/DC成分を区別せずに、統一されたツリー構造に従って共通のコンテクストを生成することに特徴がある。

【００５５】算術符号演算器から出力される符号は、固定長化処理回路22により、一つのブロック（例えば、DCT変換の単位となるブロック）の符号量が、所定のサイズに強制的に揃えられる。

【００５６】各ブロックの符号長が固定化されていることにより、ブロックを単位とした復元画像の編集が容易となる。

【００５７】一般的にいえば、符号を固定長化すると情報の欠損が生じて、復元画像の質はやや低下する。

【００５８】しかし、本発明では、画像データの情報を高精度に取得し、算術符号器で高い精度で圧縮しているため、本来的な情報の品質が高い。

【００５９】よって、固定長化処理による復元画像の品質低下を心配する必要がない。

【００６０】以上説明した、本発明の算術符号化装置の主な特徴点を図2に示す。

【００６１】像域判定・階層化処理では、画像をタイル（マクロブロック）T1～T9に分割し、各タイルに含まれる画素を背景（BG）と前景（FG）に分類する。

【００６２】タイルは、例えば、32画素×32画素のサイズであり、DCT変換の基本となるブロック（ミクロブロック：8画素×8画素）BEが16個、集まって構成される（処理1）。

【００６３】次に、完全パイプライン型算術符号化装置による符号化を行う（処理2）。算術符号化は、一つのブロック（BE）毎に行う。

【００６４】つまり、一つのブロック（BE）の符号化が終了すると、最後に符号化終了を示すEOBを付加し、算術符号器を初期化して、次のブロックの符号化を行う。

【００６５】次に、固定長化処理（処理3）を行う。これにより、一つのブロックの符号の符号長が一定の長さに揃えられる。

【００６６】符号は、符号メモリ4に一時的に蓄積される（処理４）。その後、復号化処理を経て、画像が復元される（処理5）。

【００６７】このとき、ブロックを単位として、部分的に画像の向きを変える等の処理を行う場合がある。1つのブロックの符号の符号長は一定であるため、目的とする画像ブロックの先頭画素の位置を容易に探し出すことができる。このように、復元画像の編集を容易に行うことができる。

【００６８】図3は、本発明の算術符号化装置の、より具体的な構成の一例を示すブロック図である。

【００６９】画像データは、階層分離／近似／直交変換・量子化を行う部分（階層分離／近似処理部）100に送られる。

【００７０】階層分離／近似処理部100は、タイルメモリ2000と、像域分離部2001と、特徴抽出器2002と、階層分離部2003と、BG（バックグランド）メモリ2004と、FG（フォアグランド）メモリ2005と、ビットマップメモリ2006と、直交変換器（DCT）2007と、BG近似処理器2008と、FG近似処理器2009と、量子化器2011と、をもつ。

【００７１】近似あるいは量子化された画像データと、タイルの像域判定結果を示すフラグ情報と、タイル内の各画素が背景（BG）／前景（FG）のどちらに属するかを示すビットマップデータと、近似処理が可能であったか否かを示すフラグ情報は、算術符号器（可変長符号器）200で、符号化される。

【００７２】算術符号器200におけるメモリ1006は、タイルの像域判定結果を示すフラグ情報および近似処理が可能であったか否かを示すフラグ情報を、一時的に蓄積するためのメモリである。

【００７３】また、算術符号器200の動作は、制御部1007により、統括的に制御される。

【００７４】算術符号演算器1004から出力される符号は、固定長化処理部300内に設けられている符号バッファ3002に一時的に蓄積される。この符号バッファは、１つのブロックの符号化に先立ち、オールゼロにクリアされる。

【００７５】符号読出し部3003は、所定バイト数の符号（１つのブロックの符号）が蓄積される度に、符号バッファ3002からデータを読出す。

【００７６】符号長が所定バイトに達しない場合には、

その符号の末尾に、符号バッファ3002の初期値であるゼロが付加されたデータが自動的に読み出されることになる。

【００７７】よって、ゼロを意図的に挿入して、符号を固定長化するという面倒な処理は不要となる。

【００７８】符号長カウンタは、符号長を積算し、その積算値が所定バイト数に達したかを検出して、算術符号演算器1004に符号化の終了を指示する。

【００７９】以上が、本発明の算術符号化装置の概要である。

【００８０】以下、本発明の装置の各部の具体的な構成や特徴について、図面を参照しつつ、順番に説明していく。

【００８１】まず、図4～図7を参照して、2値データのパイプライン算術符号化について説明する。

【００８２】（2値データの算術符号化処理）図4は、本発明の実施の形態１にかかる算術符号器の構成を示すブロック図である。

【００８３】図示されるように、符号器は、コンテクスト生成器700と、コンテクストテーブル（コンテクストRAM）701と、確率推定部（Qe ROM）702と、算術符号演算器703とからなり、それぞれ１クロックで１つの処理を実行する。すなわち、図4の算術符号器は、４段のパイプライン構造を有する。

【００８４】これらの構成要素の基本的な処理内容は、従来と同じである。

【００８５】本実施の形態の特徴は、Qe ROMの構成と、その周辺部分の回路構成にある。

【００８６】Qe ROM702に搭載されているテーブルの内容は図5の通りである。図5のテーブルの特徴は、従来のテーブルのデータに、さらに、“next Qe（LPS）”および“next Qe（MPS）”の各データが追加されていることである。

【００８７】これにより、図5のテーブルは、63ビットの幅を有し、従来のテーブルよりビット数が拡張されている。

【００８８】ここで、“next Qe（MPS）”とは、算術符号演算器703でMPSの符号化を行なった結果、オージェンドが初期値の1/2未満となって正規化処理が発生した場合において、コンテクストRAM701の遷移先の状態が更新され、かつ、次の符号化シンボルが前回と同じコンテクストであったために、コンテクストRAM701に対して前回と同じ番地へのアクセスが発生し、その結果、Qe ROM702の、更新された遷移先の状態に対応する番地にアクセスがなされたならば、その結果として、Qe ROM702から出力されるであろうLPSの幅（Qe）のことである。

【００８９】同じく、“next Qe（LPS）”とは、算術符号演算器703でLPSの符号化によって必然的に正規化処理が発生し、これに対応してループを回してコンテクストテーブルを更新し、同じアドレスにアクセスしたならば、Qe ROM702から出力されるであろうLPSの幅（Qe）のことである。

【００９０】つまり、正規化処理が発生し、コンテクストRAM701内のテーブルを更新して、再度、全く同じ番地にアクセスをしたならば発生するであろう、未来のQeの値を、あらかじめ、Qe ROM702内のテーブルに、現在のQeの値と併記しておくことに本実施例の特徴がある。

【００９１】図4の制御回路709には、符号化シンボルのコンテクストや算術符号演算器における正規化の発生の有無等のあらゆる情報が入力される。

【００９２】よって、制御回路709がそれらの情報から、現在のQeを選ぶか、未来のQe（MPS）あるいはQe（LPS）を選ぶかを、リアルタイムで選択することができる。このような選択を可能とするために、セレクタ706が設けられている。

【００９３】これにより、仮に正規化処理が発生しても、ループを回してテーブルを更新する間、処理を待つ必要がなく、セレクタにより未来のQe（MPS）あるいはQe（LPS）を選ぶだけでよい。よって、パイプラインに乱れが生じない。

【００９４】セレクタ704、705は、正規化処理が発生した後、次も、また正規化処理が発生する可能性があることを考慮し、前回使用したMPS値や、遷移先の状態の番号を再利用できるようにするために設けられている。

【００９５】以下、具体的に説明する。

【００９６】Qe ROM702には、状態遷移テーブルの状態番号がアドレスとして入力する。出力信号は現在の確率推定値Qe(s)（信号713）、正規化により状態遷移が起きた時の新しいQe(S)がLPS正規化とMPS正規化用に信号714、信号715として出力され、さらに、正規化による状態遷移番号が、同様に2種類（信号716 、信号717）出力され、また、MPS(S)を反転させるか否かを示すフラグ(switch-MPS 、信号718)が出力される。

【００９７】このフラグと現在の予測シンボルMPS(S)の排他的論理和をＥＯＲ回路708でとることで、新しい予測シンボルが作られる。この値とセレクタ707の出力が、正規化対象となるインデックス724となる。

【００９８】セレクタ704、705では、正規化が発生した直後の符号化シンボルが、同じコンテクストで符号化されるときには、下側の信号が選択される。すなわち、コンテクストRAM701にオーバーライトされる、更新用のMPSおよび次の状態番号が選ばれることになる。

【００９９】このような場合には、次も確率推定がはずれて、正規化処理が連続する可能性があることから、更新用のMPSと次の状態番号とを、再度、利用するようにしている。

【０１００】この場合、セレクタ705を介して出力される状態番号をアドレス変数として、Qe ROM702がアクセスされ、前回と同じ値の、Qe713、next Qe714、715が並列に出力され、符号化結果を待って、この中から一つが

選択されることになる。

【０１０１】各セレクタの選択信号は、制御回路709によって必要な状態信号を見ながら、適宜、出力される。図が煩雑となるので制御信号の詳細は省略している。

【０１０２】図5は、Qe ROMの構成例を示している。

【０１０３】上述のとおり、パイプラインが乱れるのは、Qe ROMの再読み出しが必要となることが原因である。

【０１０４】本実施例では、このような場合に、Qe ROMの読み直しが不要になるように、LPS正規化時の遷移先のQeと MPS正規化時の遷移先のQeとを同一アドレスに記憶している。ビット15からビット46がこの部分にあたる。

【０１０５】状態遷移表によれば、Qe-indexが０のときの、MPS正規化 、LPS正規化の遷移先(Qe-index)は１であることが分かる。また、Qe-indexが１のときのQeは0x2586であることが分かる。

【０１０６】こうすることによって、正規化発生直後に、同一コンテクストの符号化を行う場合も、ROMの再読み出しは不要であり、既に読み出されている複数のデータから、状況に応じて必要なものを選択するだけでよい。

【０１０７】次に、図6を参照しながら、図4の算術符号器の動作の概要を説明する。

【０１０８】第i番目のシンボルの符号化がクロックに同期して、コンテクストの生成(context det 。)、コンテクストRAMのリード(context RAM RD)、Qe ROMのリード(QeRD) 、符号化演算および正規化演算(coding/renorm)の順に処理されていく。

【０１０９】符号化シンボルとそのコンテクストは、コンテクスト生成器700で検出する 。符号化シンボルはパイプラインのタイミング調整用の遅延回路710を通して算術符号演算器703に送られる。

【０１１０】コンテクストの識別番号であるコンテクストインデックス”ｓ”は、コンテクストRAM701の入力信号となる。同時にそのコンテクストで正規化が発生する場合に、RAMの内容を更新するためのアドレス情報として、正規化インデックス712に入力される。

【０１１１】正規化インデックス712は、３クロック分の遅延を与える遅延回路である。

【０１１２】正規化インデックス712の出力信号724は、正規化処理が発生した場合に、次の遷移先の状態やMPS値を更新する際のアドレスを指定する信号（コンテクストインデックス）となる。

【０１１３】コンテクストRAM701は、リードとライトを並行して行なえるデュアルポートRAMで構成する。

【０１１４】これにより、正規化処理が生じてRAMの更新が必要となった場合でも、次のコンテクスト（図20のテンプレートを右側にずらした場合の参照画素における１と０の配置）が前回と違っていれば、符号化用のコンテクスト情報の読み出し（リード）と、正規化が生じた場合のコンテクスト情報の更新（ライト）とを、同じサイクルで、同時に実行できるようになる。

【０１１５】したがって、正規化処理が発生しても、次のコンテクストが異なっているのであれば、ROMへのアクセスの競合が生じないため、オーバーライトを待つ必要がなく、パイプラインの乱れは生じない。

【０１１６】コンテクストRAMの出力信号は、符号化シンボルの予測値MPS(S)と確率推定器の状態番号（図中state No）である。予測値MPS(S)は、パイプラインの遅延調整回路711を通って算術符号演算器703に送られる。

【０１１７】これら2つの出力信号は、セレクタ704と705に入る。制御回路709は、正規化が発生しない時は、上側の信号（つまり、コンテクストRAMから出力されるMPS値と状態番号）が選ばれるように、セレクタ704、705を制御する。

【０１１８】Qe ROM702は、セレクタ705を介して入力される状態番号をアドレス変数としてアクセスされる。

【０１１９】図5に示すように、ＱｅＲＯＭ７０２の一つのアドレスには３種類のQeが併記されているため、Qe ROM702からは、確率推定値として3種類の値が、常に出力される。そして、制御回路709が、正規化の有無やコンテクストの連続の有無を判定して、状況に応じて、それらの中の一つをリアルタイムで選択していく。

【０１２０】以下、図6を用いて、正規化処理が発生した場合の動作を説明する。

【０１２１】図6における処理801では、Qe(Si)（信号713）が選択されたものとする。ここでは第i番目のコンテクストをSiとした。

【０１２２】処理802で、第iシンボルの符号化演算と正規化処理が行われる。ここでは、LPSを符号化して正規化が発生したものとする。

【０１２３】第i+1シンボルに対しては第iシンボルの符号化演算と同一サイクルで、Qe ROM702の読み出しが実行される（処理804）。

【０１２４】第i+1シンボルのコンテクストも同じくSiであったとする。そうすると、第iシンボルでLPS正規化が起きたので、セレクタ706では、LPS正規化で遷移する先のQe値714が選択される。

【０１２５】処理805では、この値を使った符号化演算が実行される。このとき処理803ではコンテクストRAM701のコンテクストSiの内容が更新される。

【０１２６】更新処理は以下のように行われる。すなわち、セレクタ707では、LPS正規化時の次の状態番号716が選択される。一方、EOR回路708では、switch-MPS（信号718）と現在の予測値MPS(Si)（信号723）から新しい予測値（信号721）を作る。

【０１２７】上述のとおり、信号718が“１”であればMPS(Si)の値は反転する。これら2つの情報がコンテクストRAM701のアドレスSiに書かれる。

【０１２８】このときアドレスSiは、遅延回路712から信号724として出力されている。この更新処理のタイミングで、第i+2シンボルに対しては、Qe ROM702を読む必要がある。

【０１２９】この時、第i+2シンボルも同じコンテクストSiであればセレクタ704、705は下側の信号が選ばれるように制御する。その理由は、上述のとおり、処理805で正規化が再度発生するかもしれないからである。Siのコンテクスト情報の更新が終了したら 、セレクタ704 、705の上側の信号が選ばれる。

【０１３０】第i+3シンボルに対してはこのサイクルでコンテクストRAMのリードを行う（処理808）。先に説明したとおり、第i+3シンボルのコンテクストがSiと異なりSjであれば、デュアルポートROM701に対するコンテクストのリードとコンテクストSiの更新は、同時に実行される。

【０１３１】第i+3シンボルのコンテクストがSiであれば処理803でその内容が更新される。この時、セレクタ704 、705では下側の信号が選択されている。

【０１３２】このようにして正規化が発生したコンテクストで連続して符号化する場合も従来例のような無効サイクルは発生せず、パイプラインが乱れることはない。したがって、いかなる画像パターンに対しても1シンボルを１クロックで連続して符号化することができる。

【０１３３】以上説明した本発明の符号化の主要な手順をまとめると、図7に示すようになる。

【０１３４】すなわち、現在のQeと、MPS符号化で正規化処理が発生したならば出力されるであろうQe（next Qe）と、LPS符号化で正規化処理が発生したならば出力されるであろうQe（next Qe）と、を並列に出力する（ステップ30）。

【０１３５】そして、正規化が発生し、かつ次のコンテクストも同じであるときには（ステップ31）、MPSの符号化による正規化であるか、LPSの符号化による正規化であるかに応じて、未来のQe（next Qe）のうちのいずれかを選択する（ステップ32）。正規化が発生しない場合には、現在のQeを選択する（ステップ33）。

【０１３６】なお、復号化も同様に、現在のシンボルを復元中に、一つ先ないし三つ先のシンボルの復元に必要な情報を先回りして並列に入力あるいは出力しておき、復元結果が判明した時点で、それに応じて、いずれかの出力を選択する、という手法を採用することにより、算術復号処理もパイプライン化することができる。

【０１３７】しかし、符号化と同様に正規化が発生し、かつコンテクストが連続したときには、パイプラインが乱れることになる。

【０１３８】よって、符号化の場合と同様に、現在のQeと更新後のQeとを並列に出力して、実際の復号結果に応じていずれかを選択していく方式を採用する。

【０１３９】以上説明した本発明の算術符号・復号器は、図21に示すような、複合機（スキャナー、ファクシミリ装置、コピー機の機能を併せ持つ装置）に搭載するのに適している。

【０１４０】すなわち、画像を読み取って一時的にメモリに蓄積するような用途、例えばスキャナやFAX、複合機などにQM-coderを適用するには、スキャナやプリンタの高速化に伴って高速処理が要求される。

【０１４１】本発明を適用すれば、どのような画像であっても、パイプライン処理が乱れることがなく、非常に高速な符号・復号化を行なうことができる。

【０１４２】図21の複合機は、ホストプロセッサ102と、MH等の復号化回路103と、画像処理回路104と、QM符号／復号化回路105と、画像ラインメモリ106と、符号メモリ107と、モデムなどの通信インタフェース108と、スキャナなどの画像入力装置111と、プリンタなどの画像記録／表示装置112と、をもつ。

【０１４３】本発明の算術符号・復号器は、QM符号／復号化回路105に搭載される。

（多値画像の算術符号化処理）多値画像については、図10に示すような統一化された手順で、多値画像情報を2値情報に分解し、この2値情報を算術符号化する。

【０１４４】2値分解処理の基本は、あらかじめ順番が決まっている質問（判断事項）を複数、用意しておき、データが入力されると、その順番に従って質問をなし、“yes”か“no”で分岐させ、その分岐を“１”，“０”で表現することである。

【０１４５】これにより、算術符号化対象の2値シンボルが生成されることになる。復号側が、どのような順番で、どのような質問（判断）がなされるかを予め知っていれば、その符号の復号化が可能である。

【０１４６】ここで問題なのは、2値分解により得られた符号化シンボルを算術符号化する場合の、コンテクストをどのようにするかである。

【０１４７】つまり、多値データ符号化用のコンテクストを、どのように生成するかが問題となる。

【０１４８】DCT変換係数には、DC成分とAC成分とがあり、それぞれが性質が異なるので、従来は、DC成分用のコンテクストと、AC成分用のコンテクストと独立に設定しなければならない、と考えられていた。

【０１４９】つまり、図9(a)に示すように、コンテクストを切り替えるためには、DC成分（あるいはAC成分）の2進表現データと、次のDC成分（AC成分データ）の2進表現データとの境界を判定する必要がある。

【０１５０】そして、図9(b)のように、コンテクストを生成する際に、DCコンテクストのツリーからACコンテクストのツリーへの遷移が必要となる。

【０１５１】したがって、データの復元の際、１つのシンボルを復元する毎に、DC成分／AC成分の終わりを判定する処理が必要となり、１シンボル当り、最低でも2サイクルかかることになり、これで１クロックで処理を完

了させるという、完全なパイプライン処理は不可能となってしまう。

【０１５２】また、コンテクストモデルが複数あると、DC成分とAC成分の境目でパイプラインは初期状態から開始するような回路構成にならざるを得ない。

【０１５３】このような復号器では高速処理は期待できない。また、このようにDCコンテクストとACコンテクストを持つ回路構成では、回路規模は大きく、コンテクストの異なる成分間における制御は煩雑になる。

【０１５４】そこで、本発明では、敢えて、DC成分とAC成分とを区別せず、双方の成分データに対し、共通のコンテクストを割り当てる。

【０１５５】このようにすれば、図8（b）に示すように、コンテクストのツリーは、一つのツリーに統一化される。この場合、圧縮率はやや劣化するものの、図9(a)に示すような、DC成分／AC成分の終わりを判定する処理はまったく不要となる。

【０１５６】つまり、図9(a)のようにDC成分／AC成分の境界を判定する必要がなくなり、図8(a)に示すように、コンテクストを連続して生成できるようになる。

【０１５７】このように、コンテクストを共通化しても、本発明では画像情報を階層化して高精度に取得し、また、極めて高精度な圧縮をしているので、復元画像の画質が問題となることはない。

【０１５８】図10を用いて、多値画像情報の2値分解処理の具体例について説明する。

【０１５９】まず、差分データΔＶ（隣接するDCT係数の差分値）が、EOB（DCおよびAC成分を含む周波数成分ブロックの全データがゼロであることを意味する）であるかどうかを調べる（ステップ601）。その結果を符号化する。

【０１６０】もし、yesなら、“１”を“EOB”というインデックスで表現されるコンテクストで符号化して、符号化が終了する。“１”を“EOB”というインデックスで表現されるコンテクストで符号化することを、図中、1(EOB)と記す。

【０１６１】一方、ステップ601でEOBでなければ、“0”を“EOB”というインデックスで表現されるコンテクストで符号化する。このことを、図中、0(EOB)と記す。この表記は、以下、同様である。

【０１６２】このように、各ステップで判断を行い、出力が2つあるときは、右側がyesの判定、下側がnoの判定の符号化を表す。

【０１６３】従って、上述のとおり、ステップ601において、EOBであれば、１をEOBというコンテクストで符号化し符号化終了となる。そうでなければ、0をコンテクストEOBで符号化する。

【０１６４】次に、ステップ602において、差分データがΔV(V)がゼロであるかどうかを判定する。その判定結果を、コンテクスト“S0”で符号化する。ΔV(V)がゼロであれば、この時点で符号化は終了し次の数値の符号化が行われる。ΔV(V)がゼロでなければ、ステップ603に移行し、正負（+または-）の符号を符号化する。このときのコンテクストは“S1”である。

【０１６５】次に、ステップ604またはステップ605に移行する。

【０１６６】このステップ604では、差分データΔＶの絶対値が“1”より大きいか否かを判定する。つまり、Sz>0であるかを判定し、その結果をコンテクストS2で符号化する。

【０１６７】もしΔＶが“１”でなければ、ステップ606において、ΔＶの絶対値が“2”より大きいか否かを判定する。

【０１６８】つまり、Sz>１であるかを判定し、その結果を、X1というコンテクストで符号化する。

【０１６９】もし、Sz>1（ΔＶの絶対値が“2”）でなければ、ステップ607において、ΔＶの絶対値が3または4であるか、あるいは、4より大きいかを判定する。

【０１７０】つまり、Sz>3であるか否かを判定し、その結果を、コンテクストX2で符号化する。

【０１７１】ここで、Sz＝2のときは、ステップ608において、2の2進表記“10”の下位ビットの“0”をコンテクストM2で符号化する。

【０１７２】また、Sz＝3のときは、同じくステップ608において、2の2進表記“11”の下位ビットの“1”をコンテクストM2で符号化する。

【０１７３】ステップ609では、Sz>7であるかを判定し、その結果をコンテストX3で符号化する。

【０１７４】ここで、Szが4～7のとき、4，5，6，7のそれぞれの2進表記“100”，“101”，“110”，“111”の下位2ビット“00”，“01”，“10”，“11”を、コンテクストM3で符号化する（ステップ610，611）。

【０１７５】ステップ612では、Sz>15であるかを判定し、その結果をコンテクストX4で符号化する。

【０１７６】このとき、Sz=8～15のときは、それぞれの数値を2進表記して、下位３ビットをコンテクストM4で符号化する。

【０１７７】入力された差分データΔＶの値が大きい場合には、以下、同様の処理を繰り返し実行する。

【０１７８】ステップ616では、Sz>32768であるかを判定し、その結果をコンテキストX15で符号化し、Szが32768以下ならば、各数値を2進表記し、下位の数ビットをコンテクストM15で符号化する。

【０１７９】以上の説明の中で、X1～X15は、Szのmagnitude categoryを示すデータを符号化するコンテクストであり、M2～M15は、Szのmagnitude bitを符号化するためのコンテクストである。

【０１８０】以上のような順序で、多値画像情報の2値化が行われ、分解された2値データが、算術符号化の対象シンボルとなる。

【０１８１】多値データの算術符号化の全体の手順をまとめると、図11(a)に示すようになる。

【０１８２】すなわち、まず、フォーマット変換．ＥＯＢの検出を行う（ステップ810）。次に、2値分解処理を行い（ステップ811）、AC成分／DC成分を区別することなくコンテクストを生成する（ステップ812）。そして、算術符号化処理を行う（ステップ813）。

【０１８３】周波数成分の算術符号化は、図11(b)に示すような手順で行われる。ここでｋはジグザグスキャンのインデックスを表す。

【０１８４】まず、k＝0とする（ステップ801）。次に、k＝0の数値がEOBシンボルを表しているかどうかを判定する（ステップ802）。

【０１８５】ここで、k＝0時点でEOBであるということは、DC成分もAC成分もすべてゼロであるということを意味している。

【０１８６】この判定でEOBであれば、ステップ804で、１を符号化（code-1)して、そのブロックは符号化終了となる。

【０１８７】もしEOBでなければステップ803で0を符号化し、次に、ステップ805でΔV(V)の符号化を行う。

【０１８８】ステップ806の判断で1ブロック終了していなければ、ステップ807でインデックスを更新し、同様の処理を繰り返す。

【０１８９】このようにして1ブロックの符号化が終了する。ここには示していないが、周波数成分以外にDC成分として符号化する情報があれば、まず、それらを符号化する。

【０１９０】本実施の形態では、DC成分、AC成分の順に符号化する。

【０１９１】以上、算術符号化の具体例について説明した。次に、符号を固定長化する処理について説明する。

【０１９２】（符号の固定長化処理の内容）以下、図12～図16を参照して、一つのブロックの符号量を一定のサイズに揃える処理について説明する。

【０１９３】図12(a)は、一つのブロックの符号量を一定のサイズに揃えるための回路の構成を示すブロック図である（図3に示される回路と同じ構成である）。

【０１９４】算術符号器200から出力される符号は、固定長化処理部300内に設けられている符号バッファ3002に一時的に蓄積される。この符号バッファ3002は、１つのブロックの符号化に先立ち、オールゼロにクリアされる。

【０１９５】符号読出し部3003は、所定バイト数の符号（１つのブロックの符号）が蓄積される度に、符号バッファ3002からデータを読出す。

【０１９６】符号長が所定バイトに達しない場合には、その符号の末尾に、符号バッファ3002の初期値であるゼロが付加されたデータが自動的に読み出されることになる。

【０１９７】よって、ゼロを意図的に挿入して、符号を固定長化するという面倒な処理は不要となる。

【０１９８】符号長カウンタ3001は、符号長を積算し、その積算値が所定バイト数に達したかを検出して、算術符号演算器1004に符号化の終了を指示する。

【０１９９】図12(b)，図12(c)は、一つのブロックに関するDCT係数を符号化した場合の符号量の一例を示している。

【０２００】図12(b)に示されるように、符号化の途中で、積算の符号長が所定のバイト数が超えた場合には、EOBを付加して符号化処理を打ち切り、所定のサイズのデータを読み出す。最後のDCT変換係数に対応する符号ｃｉの、はみ出てた部分（Ｆ：図中、斜線で示される）を破棄する。

【０２０１】一方、図12(c)のように、積算の符号長が所定バイト数に収まる場合には、末尾に、自動的にゼロがパディングされて（符号バッファ3002の初期データであるゼロが挿入されたことになる）、所定バイトの符号が読みだされる。

【０２０２】次に、図13を参照して、符号量を一定にする処理を含む、算術符号化の手順を説明する。

【０２０３】まず、ブロックの符号化に先立って符号バッファ3002をクリアする（ステップ501）。

【０２０４】次に、ステップ502によって、コンテクストメモリ（図4の参照符号7001）や算術符号演算器（図4の参照符号703）のレジスタをクリアする。

【０２０５】コンテクストメモリは、算術符号器内部の確率推定器の学習効果を早めるために、クリアせずに、トレーニングデータによる初期値を設定するようにしてもよい。

【０２０６】次に、EOBであるか否かを判定する（ステップ503）。

【０２０７】次に、符号化バイト数が所定値を超えたかどうか判定する（ステップ504）。所定値に満たなければ、係数の符号化を行う（ステップ505）。この際に符号化バイト数を計数する。これらの処理を1ブロック64個のＤＣＴ係数の符号化が終わるまで、または終了条件が満たされるまで(ステップ506)、繰り返す。

【０２０８】符号化終了時には、EOBシンボルを符号化し（ステップ507)、算術符号器の符号レジスタの記憶データを吐き出して（ステップ508）、1ブロックの符号化が終了する。

【０２０９】なお、図13のステップ504において、所定条件が満たされたとき、EOBを符号化して符号化を終了するのは、符号化を高速化するためである。これらの符号は、符号バッファ3002に蓄積される。符号読出し部3003は、符号バッファ3002から、所定バイト数の符号をバッファ先頭から読み出す。

【０２１０】例えば、所定バイト数が８バイトとすると、簡単な画像ブロックでは、B0;B1;B2;0;0;0;0;0のよ

うに符号バイトB0,...,B2の後にゼロが連なる。簡単ブロックでは符号は2バイト程度である。

【０２１１】一方、複雑なブロックでは、B0;B1;B2;B3;B4;B5;B6;B7のように、符号バイトが連続する。最後のB7は符号の途中で打ち切られる（図12(b)，図12(c)）。

【０２１２】簡単な画像では、図12(c)のように、EOB情報も含めて所定バイト内に収まり、その後にゼロがパディングされる。

【０２１３】複雑な画像ブロックでは，DCT係数Ciの途中の情報は、符号として復号器に送られることはなく捨てられる（図12(a)）。このようにして、算術符号を使ったブロック固定サイズの符号化が行われる。

【０２１４】次に、復号化動作を図14～図15を用いて説明する。

【０２１５】図14は、固定長化処理がなされた符号を受信して、復号する算術復号器の構成を示すブロック図である。

【０２１６】以下、算術復号器の周辺を中心に説明する。

【０２１７】符号データは符号入力部310によって、外部にある符号メモリから符号バッファ309に所定バイト数の符号を入力する。1ブロックの復号化は、この符号バッファに入力された符号のみを使う。セレクタ307は符号データを入力するか、ゼロを入力するかを選択する。

【０２１８】符号長カウンタ308は算術復号器306が読み込んだ符号バイト数を計数し、それが所定バイト数311を超えたら、セレクタ307でゼロ313を算術復号器306に入力するように選択信号を出す。

【０２１９】算術復号化は符号化同様に2つのレジスタを使う。一つは符号レジスタであるCレジスタ、もう一つは数直線の幅を表すAレジスタである。本実施例ではどちらも16ビットとしている。これら2つのレジスタの大小比較によってシンボルの復号を行うことができる。

【０２２０】図16は、復号化の手順を示すフロー図である。ステップ701では、係数メモリ305をクリアする。ステップ702では、符号バッファ309に所定バイト数の符号を入力する。

【０２２１】ステップ703では、符号化同様にコンテクストメモリ304と、算術復号化に使う2つのレジスタ（CレジスタとAレジスタ）をクリアする。

【０２２２】ステップ704では、シンボルを復元し、ステップ705では、復元シンボルがEOBかどうかを判定する。そうでなければステップ706で数値の復号化を行う。

【０２２３】シンボル復号化の際には入力した符号バイト数を計数する。ステップ706が終了すると、１つの係数が復元できる。ステップ707では、復号化した符号バイト数が所定値に達したかどうかを判断し、そうであればセレクタの入力信号を切り替え、それ以降は、ゼロが算術復号器に入るようにする(ステップ708)。

【０２２４】ステップ709は、Cレジスタに読み込んだ符号バイト数が所定値よりも3バイト多くなったかどうか判断する。もしそうであれば、所定バイト数の復号化を終了とする。

【０２２５】最後に復号した係数は符号データが完結していないので、ステップ710で、最終係数をゼロとして1ブロックの復号を終える。

【０２２６】図15は、復号化の様子を示す図である。

【０２２７】Cレジスタは16ビットであるが、符号の入力バッファを1バイト持つ構成とする。ブロックの復号化の開始時点では、3バイトの符号が符号バッファを含むCレジスタに読み込まれているものとする。

【０２２８】シンボルの復号化に伴って、Cレジスタは左にシフトされ、符号バッファが空になると新しい符号バイトを読み込む。

【０２２９】係数Ci-1を復号し終わった時点では全ての符号の復号が終わっておらず、次の係数Ciを復号する。

【０２３０】係数Ciの復号化が終わると、符号バッファには、所定バイト数プラス３バイトの符号が読み込まれており、所定バイト数の復号化が終わったことが分かる。

【０２３１】最後の係数Ciの符号は途中で分断されているので正しく復号できない。したがって、最後に復号した係数をゼロとする。

【０２３２】（カラー画像における処理）図17は、本発明の固定長化処理を行う機能をもつ符号器を搭載した、カラー複合機の要部の構成を示すブロック図である。

【０２３３】カラー画像は、ブロック（8×8画素）を単位として分割され、各ブロックは、カラーブロックメモリ901に蓄積される。

【０２３４】色変換器902はRGBのカラー画像を輝度と色差成分に分離する。

【０２３５】ここではYCbCrの色空間とする。分解されたカラー成分はそれぞれのメモリ903～905に格納される。

【０２３６】通常、色差成分は緩やかな信号で輝度成分よりも情報量は少ない。そこで、輝度成分と色差成分は異なる符号長に圧縮するものとする。

【０２３７】レジスタ909～911は輝度成分と色差成分の所定バイト数を設定するレジスタである。これらの値は制御部912によって設定される。参照符号907は、本発明のブロック固定長処理機能付きの算術符号器である。

【０２３８】セレクタ906とセレクタ908はカラー成分とその符号化バイト数を符号器907に供給する。セレクタの選択信号は制御部912から出力される。

【０２３９】このように構成すると、例えば（Y，Cb，Cr）＝（8，4，4）バイトや（Y，Cb，Cr）＝（12，2，2）バイトなど符号バイト数の組み合わせを自由に選択できる。

【０２４０】ブロック毎に固定サイズなので、部分復号化が容易である。また、1ページ復元しなくても符号データのままで回転処理を行うことができるなど、画像編集が容易となる。

【０２４１】また、メモリコストを削減できる。また、符号長の選択によって画質調整も容易となる。

【０２４２】以上の実施の形態では、画像ブロックを8×8画素とし、これはDCTの変換サイズと同じにしてあるが、画像ブロックを例えば32×32画素のように大きくとってもよい。この方が、学習機能を持つ算術符号では圧縮性能が高まるので画質が高くとれる。

【０２４３】このように、符号長を所定サイズに揃える機能ブロックをもつことにより、予め定めた大きさの画像ブロック毎に、一定サイズの符号を出力することが可能となる。

【０２４４】ここで、画像ブロックを圧縮する符号は効率の高い可変長符号であり、従来よりも高い画質を実現できる。

【０２４５】また、ブロックの符号長を可変にできるので画質調整も容易である。ディジタル複合機の画像処理に連携して、少ないメモリで画像編集ができる。

【０２４６】

【発明の効果】以上説明したように本発明によれば、2値画像および多値画像の双方について、柔軟に、しかも、ほとんど算術符号化アルゴリズムで決まる限界の速度でもって算術符号化･算術復号化することができる。

【０２４７】しかも、2値画像と多値画像を区別することなく、共通のコンテクストを用いて符号・復号化ができるため、符号・復号器の構成もきわめて簡素化される。

【０２４８】また、一つのブロック（処理単位）の符号長を所定の長さに揃えることにより、ブロックを単位とした復元画像の編集も容易に行える。

【図面の簡単な説明】

【図１】本発明の算術符号化装置の一例の全体構成を示す図

【図２】図１の算術符号化装置の特徴を説明するための図

【図３】本発明の算術符号化装置の具体的な構成の一例を示すブロック図

【図４】本発明のハザードレスパイプライン算術符号器の全体構成を示すブロック図

【図５】本発明における確率推定メモリの構成例を示す図

【図６】図４の算術符号器のパイプライン動作を説明するためのタイミング図

【図７】図４の算術符号器の特徴的な動作を説明するためのフロー図

【図８】（ａ）本発明における、DC成分用コンテクストとAC成分用コンテクストの切り替えを説明するための図
（ｂ）本発明における、統一されたコンテクストのツリーを示す図

【図９】（ａ）従来例における、DC成分用コンテクストとAC成分用コンテクストの切り替えを説明するための図
（ｂ）従来例における、DC成分用コンテクストのツリーからAC成分用コンテクストのツリーへの遷移を示す図

【図１０】本発明において、多値画像データを2値分解して符号化シンボルを生成する処理の手順を示す図

【図１１】（ａ）多値データの符号化処理の手順を示すフロー図
（ｂ）周波数成分の符号化処理の手順を示すフロー図

【図１２】（ａ）１ブロックの符号量を所定のサイズに揃える回路の構成を示すブロック図
（ｂ）複雑な多値画像を符号化した場合における、符号量の増大の様子を示す図
（ｃ）単純な多値画像を符号化した場合における、符号量の増大の様子を示す図

【図１３】１ブロックの符号化（固定長化処理を含む）の手順を示すフロー図

【図１４】固定長化処理を経た符号を復元する算術復号器の構成を示すブロック図

【図１５】図１４の算術復号器の復号動作を説明するための図

【図１６】図１４の算術復号器の復号動作の手順を示すフロー図

【図１７】本発明の固定長化処理機能付きの算術符号器を搭載した、カラー画像の処理装置の構成を示すブロック図

【図１８】一般的な算術符号器の基本構成を示す図

【図１９】算術符号化の原理を説明するための図

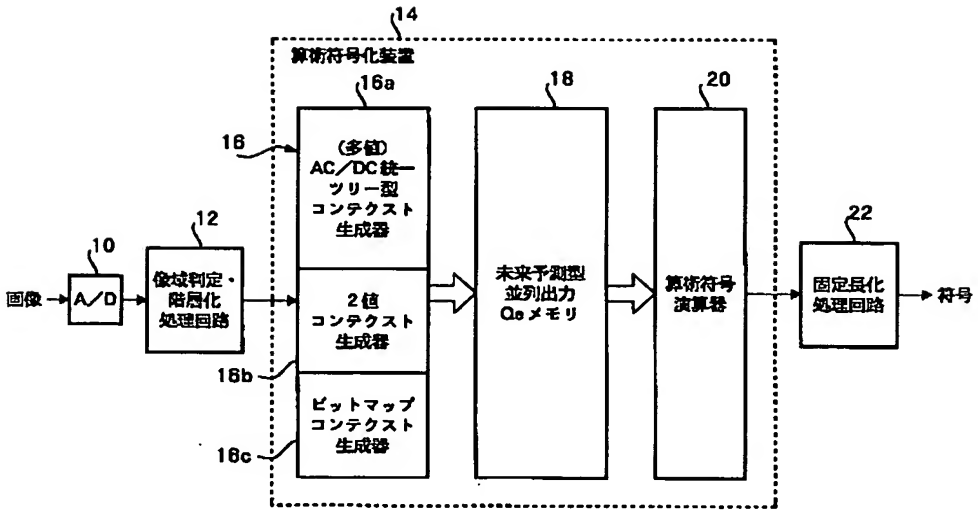
【図２０】JBIGにおける算術符号用テンプレートを示す図

【図２１】本発明の算術符号化装置を搭載した画像処理装置の構成を示すブロック図

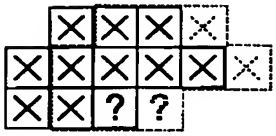
【符号の説明】

１０　Ａ／Ｄ変換器
１２　像域判定・階層化処理回路
１４　算術符号化装置
１６　コンテクスト生成器
１８　未来予測型並列出力Qeメモリ
２０　算術符号演算器
２２　固定長化処理回路

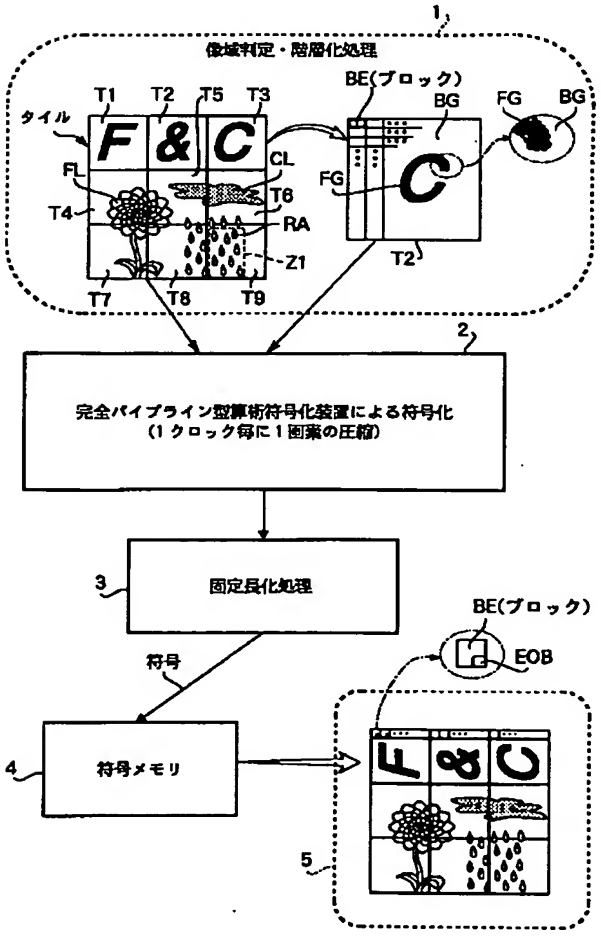
【図１】



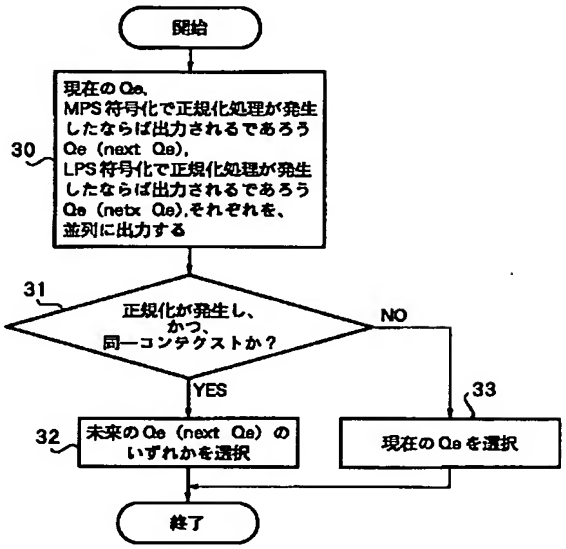
【図２０】



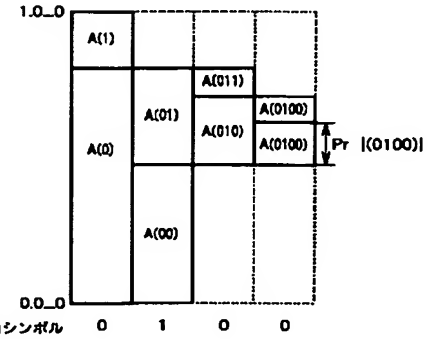
【図２】



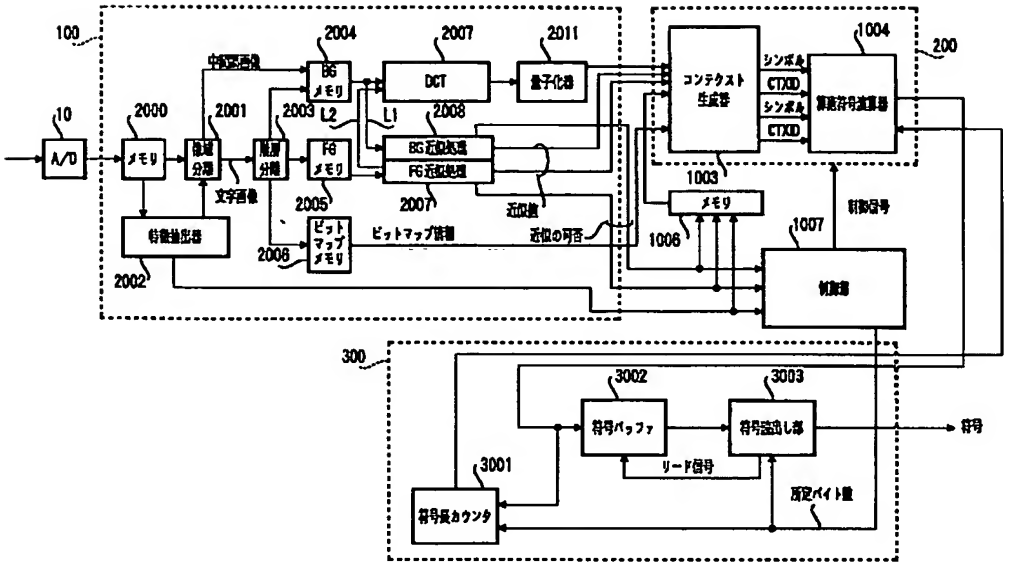
【図７】



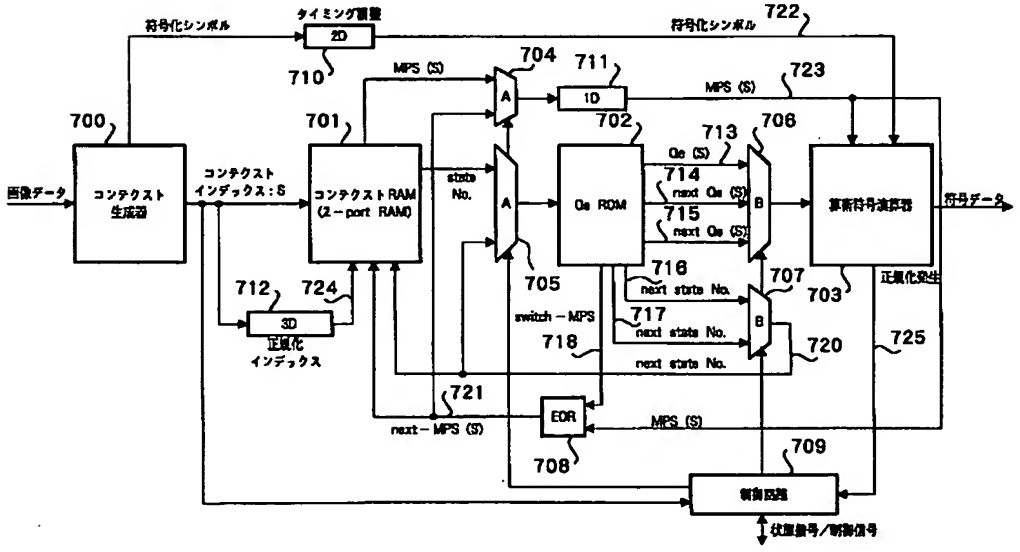
【図１９】



【図３】


100
10
A/D
2000
メモリ
2001
領域分離
文字画像
2002
特徴抽出器
2003
階層分離
2004
BG メモリ
中間画像
2005
FG メモリ
2006
ビットマップメモリ
ビットマップ情報
2007
DCT
2008
BG近似処理
FG近似処理
近似度
近似の可否
2011
量子化器
コンテクスト生成器
シンボル
CTXID
シンボル
CTXID
1004
算術符号演算器
200
1003
メモリ
1006
1007
制御器
制御信号
300
3002
符号バッファ
3003
符号読出し部
符号
リード信号
3001
符号長カウンタ
設定バイト数

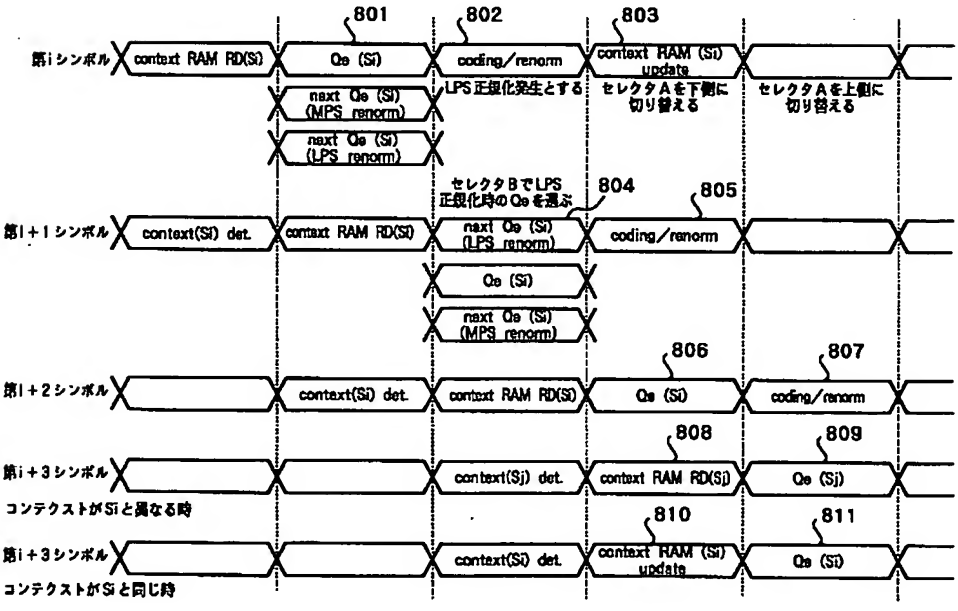
【図４】


符号化シンボル
タイミング調整
2D
710
MPS (S)
704
A
711
1D
符号化シンボル
722
MPS (S)
723
700
画像データ
コンテクスト生成器
コンテクストインデックス：S
701
コンテクストRAM (2－port RAM)
state No.
A
702
Qe ROM
713
Qe (S)
714
next Qe (S)
715
next Qe (S)
706
B
算術符号演算器
符号データ
716
next state No.
707
703
正規化発生
712
724
3D
正規化インデックス
705
switch－MPS
717
next state No.
B
718
next state No.
720
725
721
next－MPS (S)
EOR
MPS (S)
708
709
制御回路
状態信号/制御信号

【図５】

アドレス	62 Qe	46 next Qe (LPS)	30 next Qe (MPS)	14 next state (LPS)	7 next state (MPS)	1 0 switch
0	0x5a1d	0x2586	0x2586	0000001	0000001	1
1	0x2586	0x5a7f	0x1114	0001110	0000010	0
2	0x1114	0x17b9	0x080b	0010000	0000011	0
⋮						
112	0x59eb	0x59eb	0x5522	1110000	1101111	1

【図６】

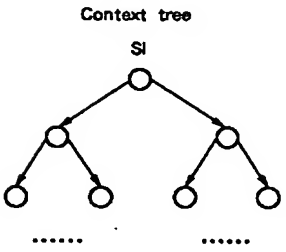


【図８】

(a)

	DC成分			AC成分			
シンボル	T_{00}, T_{01}, ..., T_{0i}			T_{10}, T_{11}, ..., T_{1i}			.....
コンテクスト	S_{00}, S_{01}, ..., S_{0i}			S_{10}, S_{11}, ..., S_{1i}			.....

(b)

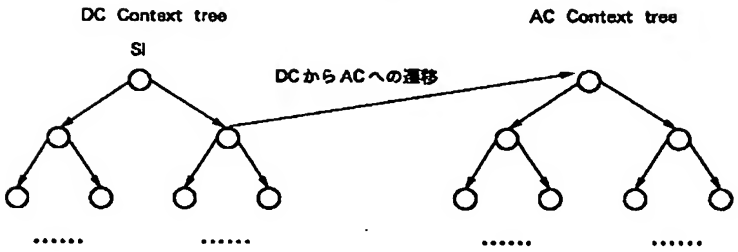


【図９】

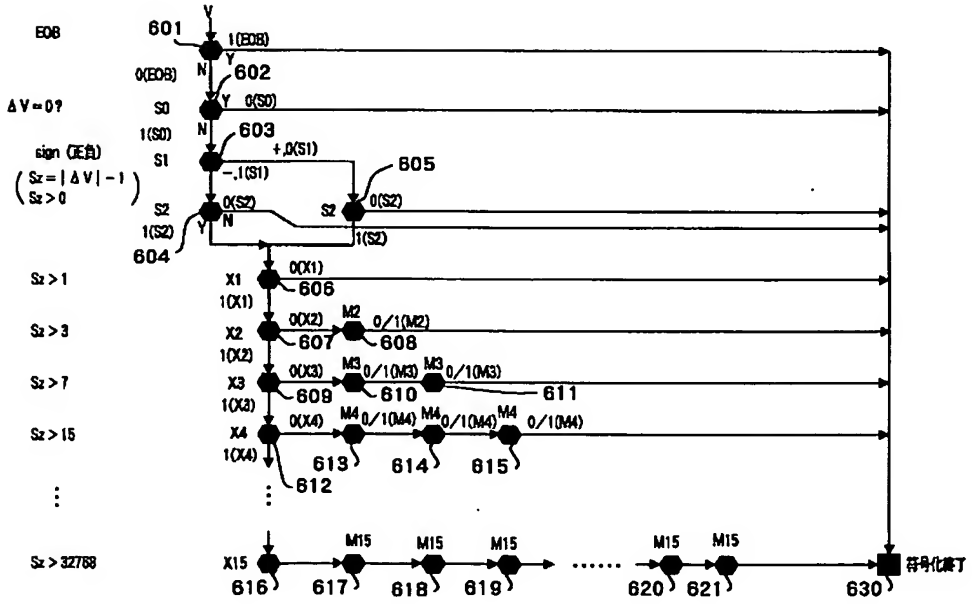
(a)

	DC成分			AC成分			
シンボル	T_{00}, T_{01}, ..., T_{0i}			T_{10}, T_{11}, ..., T_{1i}			.....
コンテクスト	S_{00}, S_{01}, ..., S_{0i}			S_{10}, S_{11}, ..., S_{1i}			.....

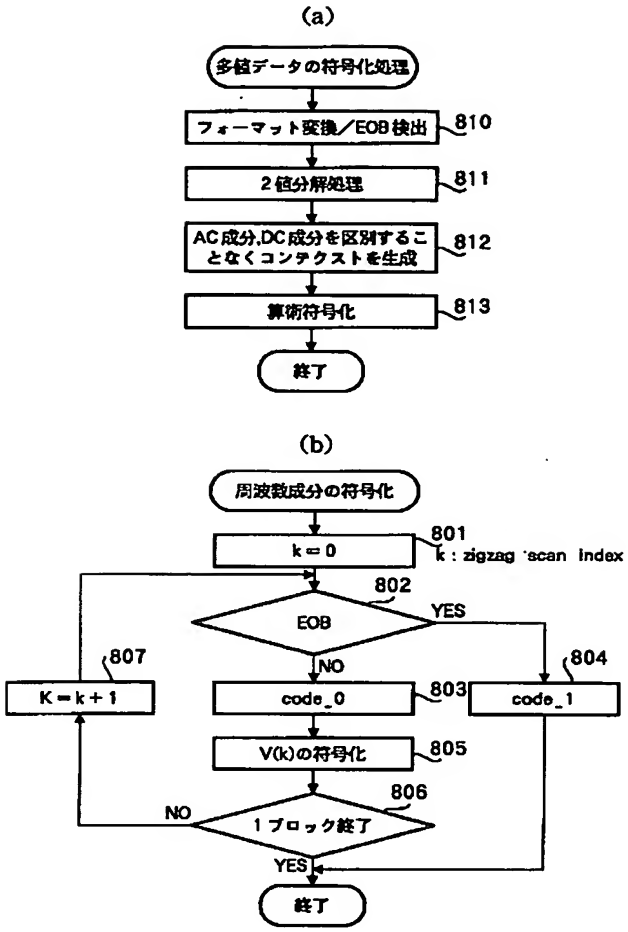
(b)



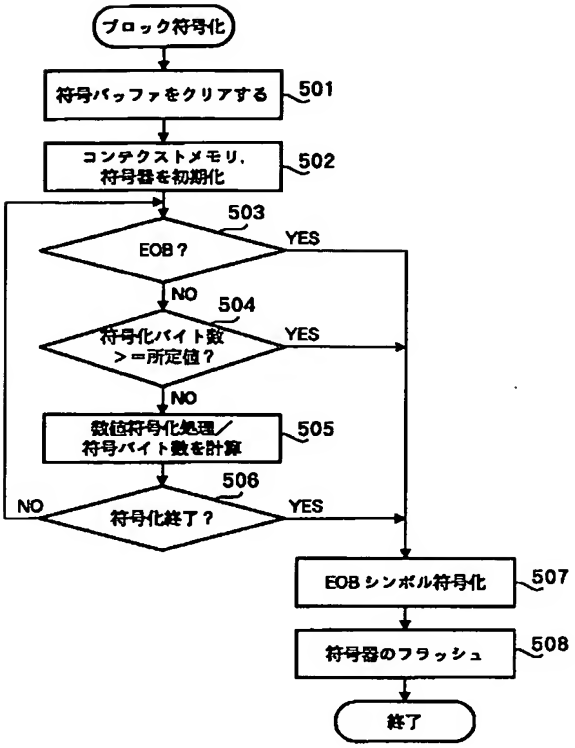
【図１０】



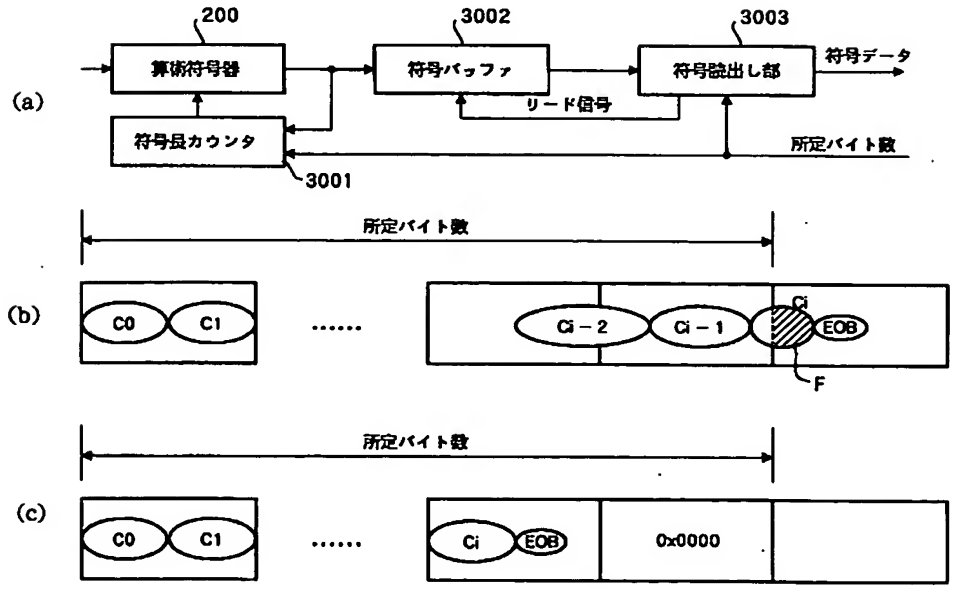
【図１１】


(a)
多値データの符号化処理
フォーマット変換／EOB検出
810
2値分解処理
811
AC成分,DC成分を区別することなくコンテクストを生成
812
算術符号化
813
終了
(b)
周波数成分の符号化
k＝0
801
k : zigzag scan index
802
EOB
YES
NO
807
804
K＝k＋1
code_0
803
code_1
V(k)の符号化
805
806
NO
1ブロック終了
YES
終了

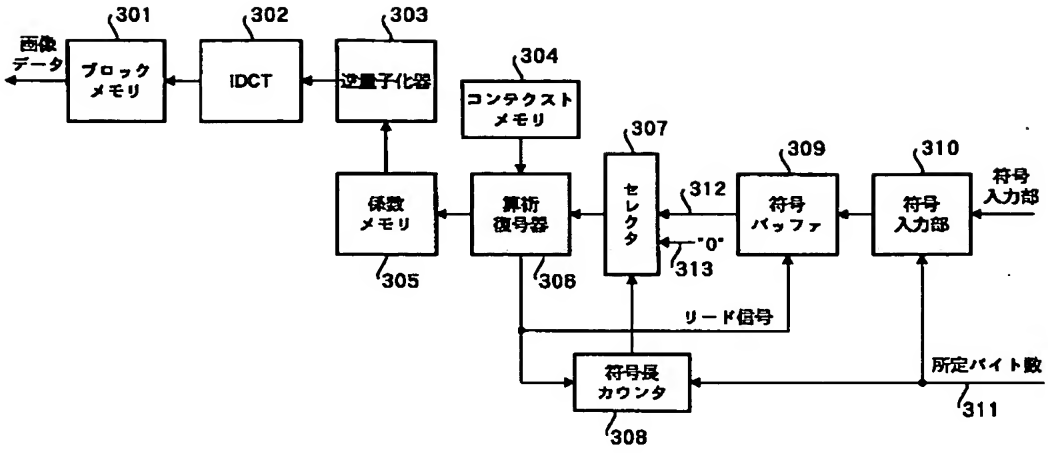
【図１３】


ブロック符号化
符号バッファをクリアする
501
コンテクストメモリ,符号器を初期化
502
503
EOB？
YES
NO
504
符号化バイト数＞＝所定値？
YES
NO
数値符号化処理／符号バイト数を計算
505
506
NO
符号化終了？
YES
EOBシンボル符号化
507
符号器のフラッシュ
508
終了

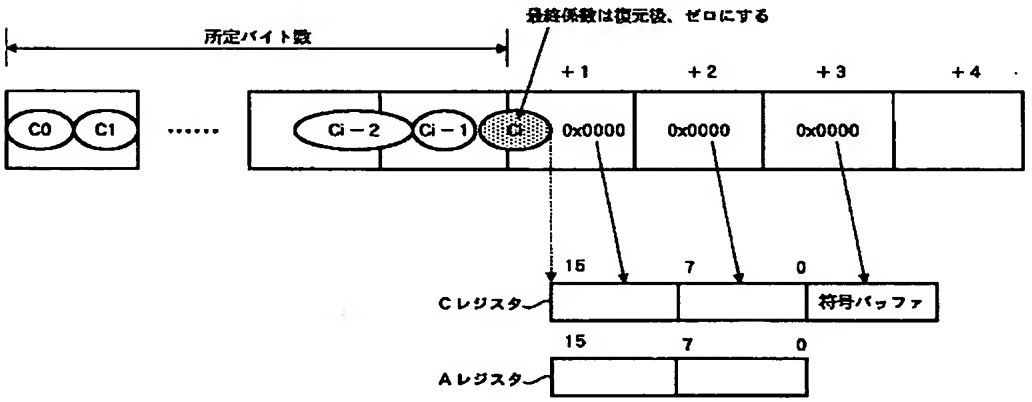
【図１２】


(a)
200
算術符号器
3002
符号バッファ
3003
符号読出し部
符号データ
リード信号
符号長カウンタ
所定バイト数
3001
所定バイト数
(b)
C0
C1
.....
Ci－2
Ci－1
Ci
EOB
F
所定バイト数
(c)
C0
C1
.....
Ci
EOB
0x0000

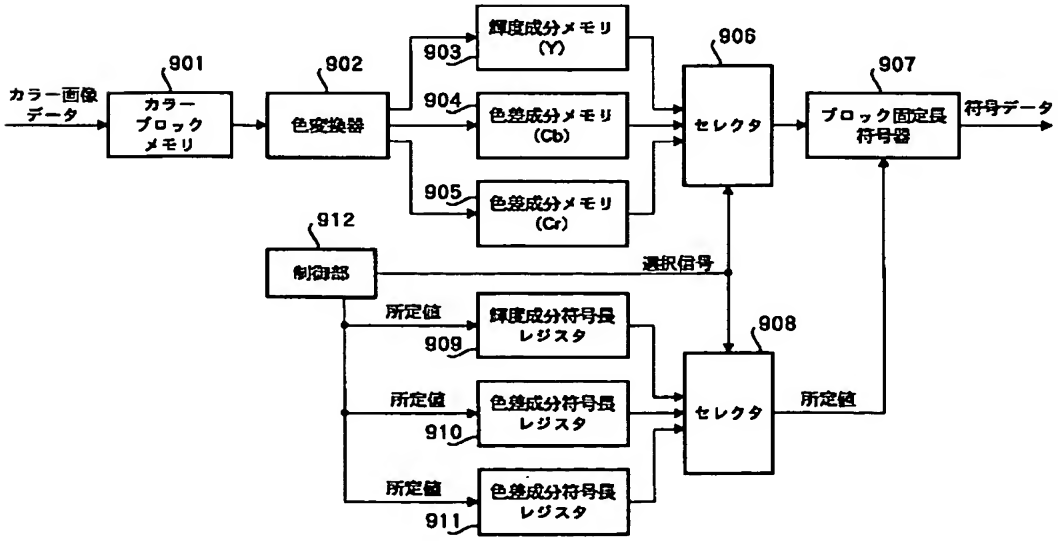
【図１４】


画像データ
301
ブロックメモリ
302
IDCT
303
逆量子化器
304
コンテクストメモリ
307
セレクタ
309
符号バッファ
310
符号入力部
符号入力部
312
"0"
313
係数メモリ
305
算術復号器
306
リード信号
符号長カウンタ
308
所定バイト数
311

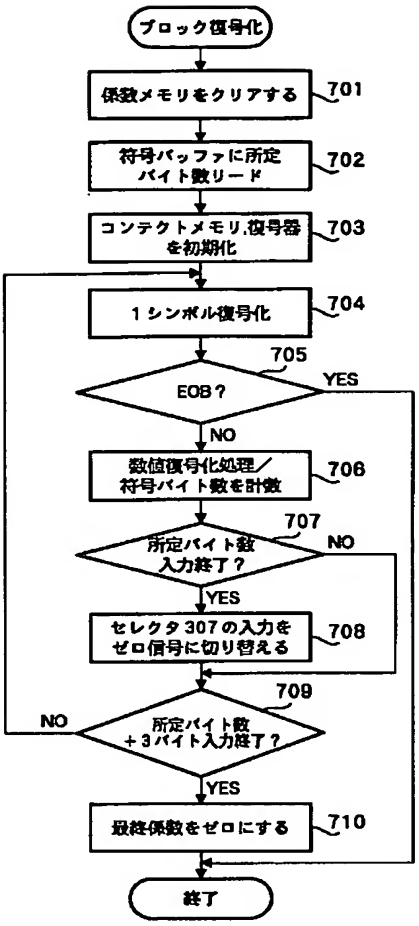
【図１５】


所定バイト数
最終係数は復元後、ゼロにする
+1
+2
+3
+4
C0
C1
Ci－2
Ci－1
Ci
0x0000
0x0000
0x0000
15
7
0
Cレジスタ
符号バッファ
15
7
0
Aレジスタ

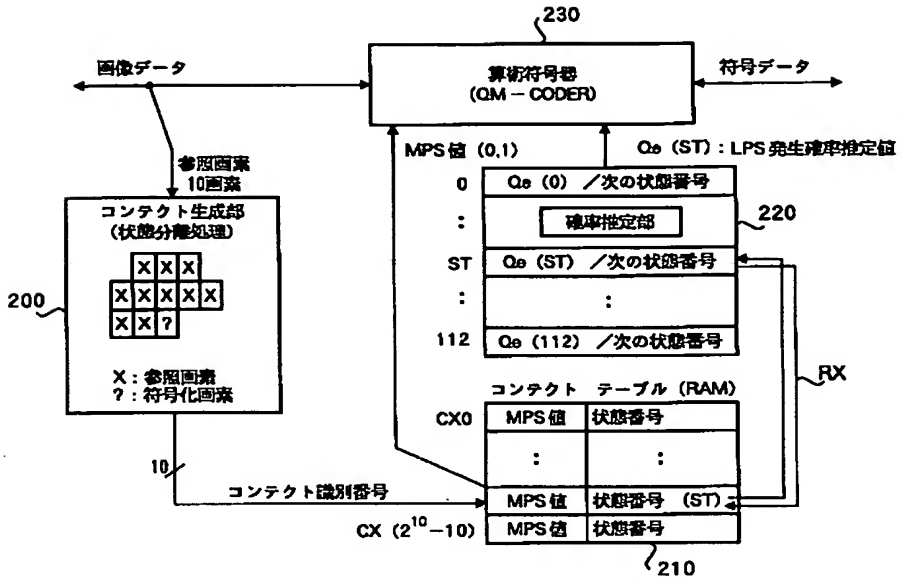
【図１７】


カラー画像データ
901
カラーブロックメモリ
902
色変換器
903
輝度成分メモリ(Y)
904
色差成分メモリ(Cb)
905
色差成分メモリ(Cr)
906
セレクタ
907
ブロック固定長符号器
符号データ
912
制御部
選択信号
所定値
909
輝度成分符号長レジスタ
908
所定値
910
色差成分符号長レジスタ
セレクタ
所定値
所定値
911
色差成分符号長レジスタ

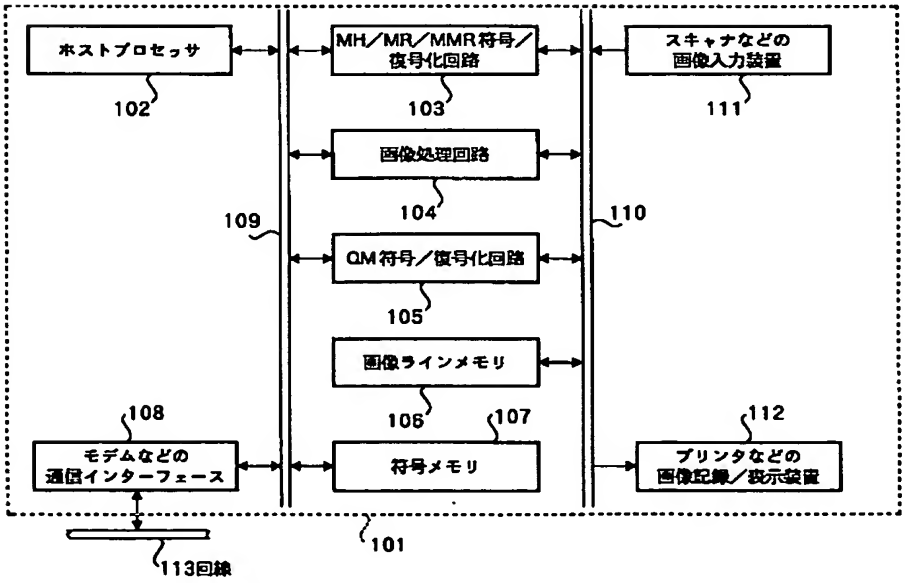
【図１６】


ブロック復号化
係数メモリをクリアする
701
符号バッファに所定バイト数リード
702
コンテクストメモリ,復号器を初期化
703
１シンボル復号化
704
705
EOB？
YES
NO
数値復号化処理／符号バイト数を計数
706
707
所定バイト数入力終了？
NO
YES
セレクタ307の入力をゼロ信号に切り替える
708
709
所定バイト数＋３バイト入力終了？
NO
YES
最終係数をゼロにする
710
終了

【図１８】


230
画像データ
算術符号器
(QM－CODER)
符号データ
MPS値 (0,1)
Qe (ST)：LPS発生確率推定値
参照画素
10画素
コンテクト生成部
(状態分離処理)
200
X：参照画素
？：符号化画素
0
Qe (0) ／次の状態番号
確率推定部
220
ST
Qe (ST) ／次の状態番号
112
Qe (112) ／次の状態番号
RX
コンテクト テーブル (RAM)
CX0
MPS値
状態番号
10
コンテクト識別番号
MPS値
状態番号 (ST)
CX (2^10－10)
MPS値
状態番号
210

【図２１】


ホストプロセッサ
102
MH／MR／MMR符号／復号化回路
103
スキャナなどの画像入力装置
111
画像処理回路
104
109
110
QM符号／復号化回路
105
画像ラインメモリ
106
107
108
モデムなどの通信インターフェース
符号メモリ
112
プリンタなどの画像記録／表示装置
101
113回線

フロントページの続き

Ｆターム(参考) 5C059 KK14 LA10 MA23 MC11 MC24
MC32 MC34 MC38 ME11 PP20
PP29
5C078 AA04 BA35 BA57 CA26 CA31
DA01 DA21
5J064 AA03 BA10 BA16 BB06 BC01
BC03 BC16 BC17 BC25 BC28
BD06 BD07